**First project - directions**

name: Build and deploy to GitHub Pages

on:

push:

branches: [main]

jobs:

build-and-deploy:

runs-on: ubuntu-latest

steps:

- name: Checkout 🛎️

uses: actions/checkout@v2.3.1

- name: Install and build 🔧

run: |

npm install

npm run build

- name: Deploy 🚀

uses: JamesIves/github-pages-deploy-action@4.1.0

with:

branch: gh-pages

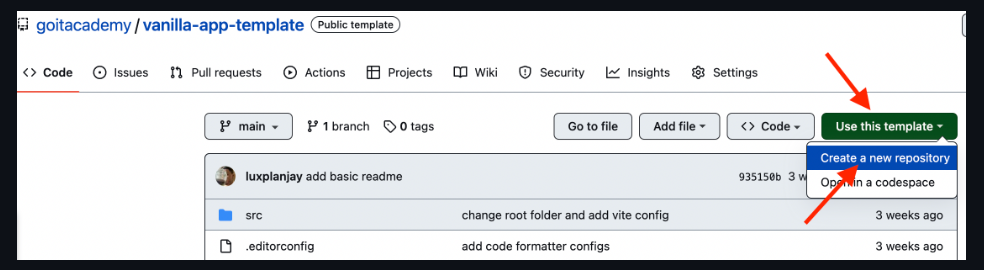
folder: dist

# **Vanilla App Template**

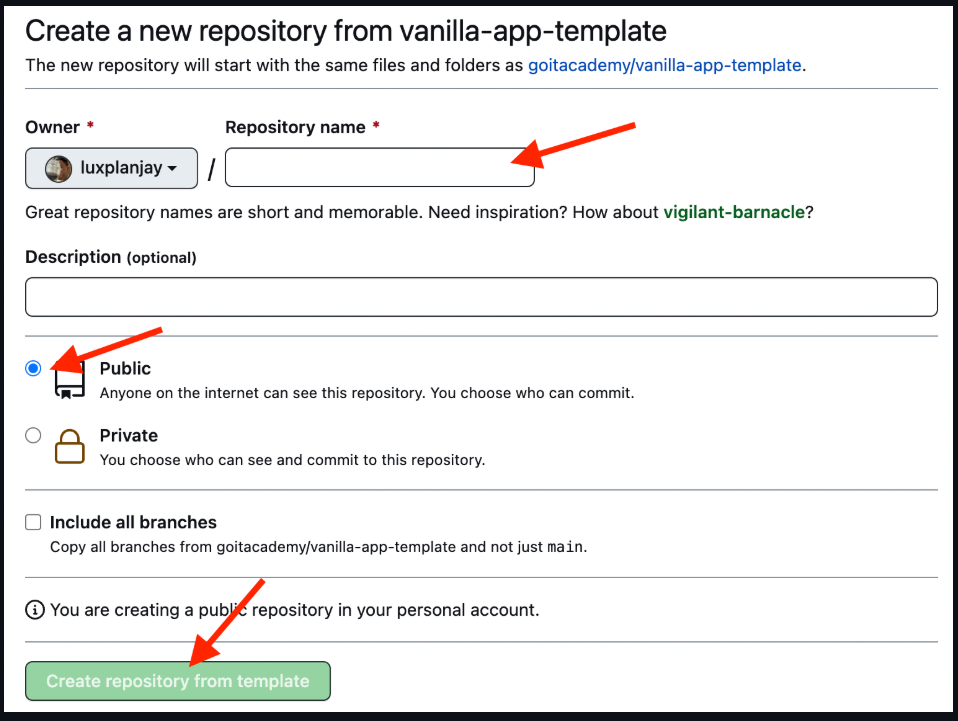
Цей проект було створено за допомогою Vite. Для знайомства та налаштування додаткових можливостей [звернись до документації](https://vitejs.dev/).

## **Створення репозиторію за шаблоном**

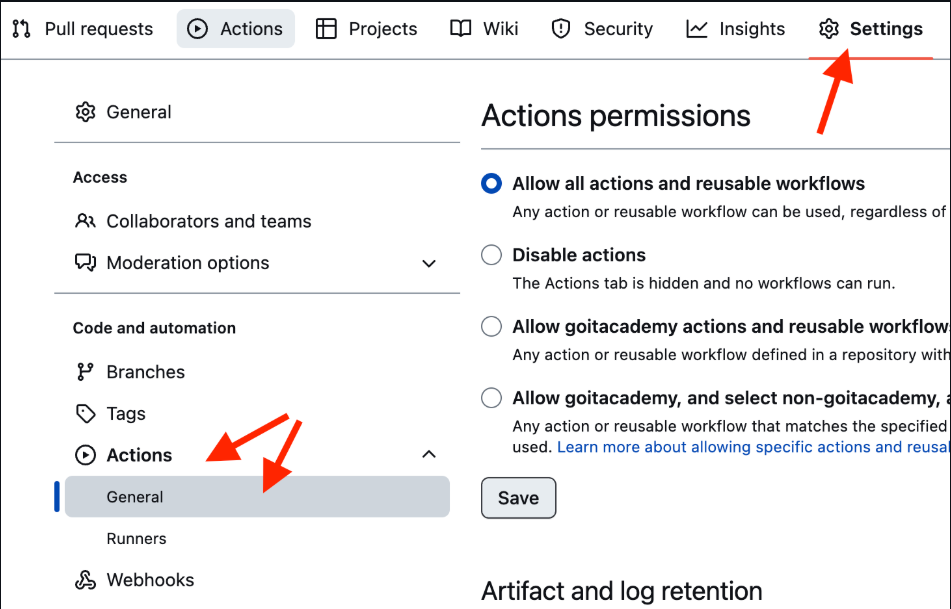
Використовуй цей репозиторій організації GoIT як шаблон для створення репозиторію свого проекту. Для цього натисни на кнопку «Use this template» і обери опцію «Create a new repository», як показано на зображенні.



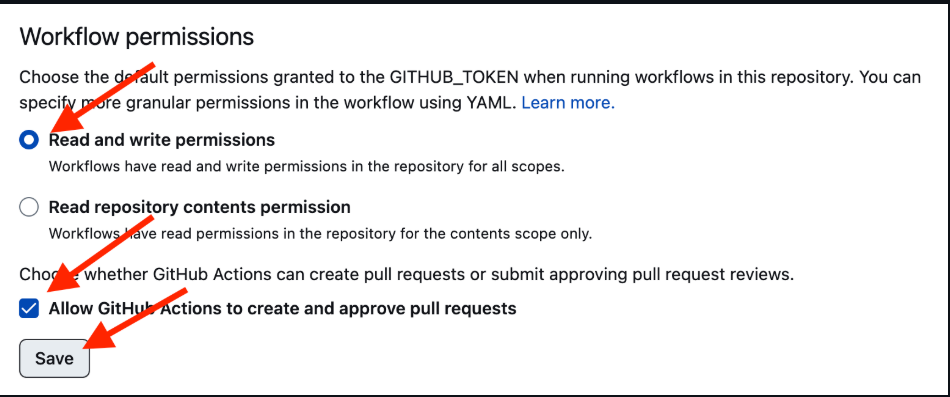
На наступному етапі відкриється сторінка створення нового репозиторію. Заповни поле його імені, переконайся, що репозиторій публічний, після чого натисни кнопку .



Після того, як репозиторій буде створено, необхідно перейти в налаштування створеного репозиторію на вкладку Settings > Actions > General як показано на зображенні.



Проскроливши сторінку до самого кінця, в секції «Workflow permissions» обери опцію «Read and write permissions» і постав галочку в чекбоксі. Це необхідно для автоматизації процесу деплою проекту.



Тепер у тебе є особистий репозиторій проекту, зі структурою файлів та папок репозиторію-шаблону. Далі працюй з ним, як з будь-яким іншим особистим репозиторієм, клонуй його собі на комп'ютер, пиши код, роби коміти та відправляй їх на GitHub.

## **Підготовка до роботи**

1. Переконайся, що на комп'ютері встановлено LTS-версію Node.js. [Скачай та встанови](https://nodejs.org/en/) її якщо необхідно.
2. Встанови базові залежності проекту в терміналі командою npm install.
3. Запусти режим розробки, виконавши в терміналі команду npm run dev.
4. Перейдіть у браузері за адресою [http://localhost:5173](http://localhost:5173/). Ця сторінка буде автоматично перезавантажуватись після збереження змін у файли проекту.

## **Файли і папки**

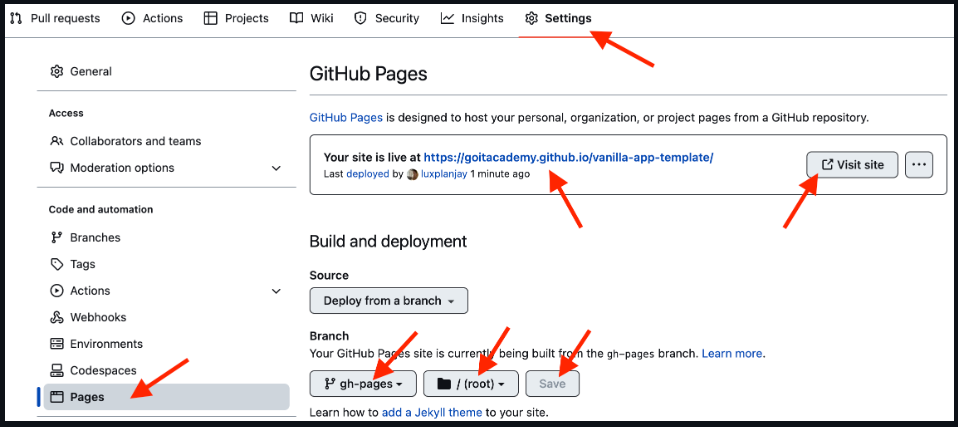
* Файли розмітки компонентів сторінки повинні лежати в папці src/partials та імпортуватись до файлу index.html. Наприклад, файл з розміткою хедера header.html створюємо у папці partials та імпортуємо в index.html.
* Файли стилів повинні лежати в папці src/css та імпортуватись до HTML-файлів сторінок. Наприклад, для index.html файл стилів називається index.css.
* Зображення додавай до папки src/img. Збирач оптимізує їх, але тільки при деплої продакшн версії проекту. Все це відбувається у хмарі, щоб не навантажувати твій комп'ютер, тому що на слабких компʼютерах це може зайняти багато часу.

## **Деплой**

Продакшн версія проекту буде автоматично збиратися та деплоїтись на GitHub Pages, у гілку gh-pages, щоразу, коли оновлюється гілка main. Наприклад, після прямого пуша або прийнятого пул-реквесту. Для цього необхідно у файлі package.json змінити значення прапора --base=/<REPO>/, для команди build, замінивши <REPO> на назву свого репозиторію, та відправити зміни на GitHub.

**"build": "vite build --base=/<REPO>/",**

Далі необхідно зайти в налаштування GitHub-репозиторію (Settings > Pages) та виставити роздачу продакшн версії файлів з папки /root гілки gh-pages, якщо це не було зроблено автоматично.

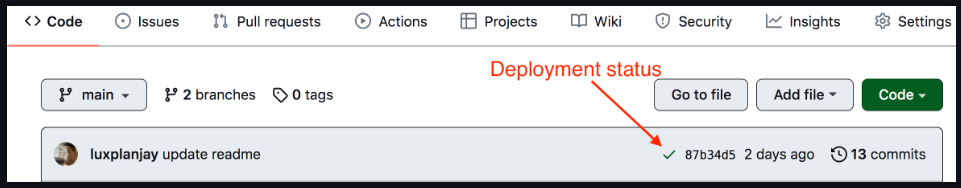


## **Статус деплою**

Статус деплою крайнього коміту відображається іконкою біля його ідентифікатора.

* ****Жовтий колір**** - виконується збірка та деплой проекту.
* ****Зелений колір**** - деплой завершився успішно.
* ****Червоний колір**** - під час лінтингу, збірки чи деплою сталася помилка.

Більш детальну інформацію про статус можна переглянути натиснувши на іконку, і в вікні, що випадає, перейти за посиланням Details.



## **Жива сторінка**

Через якийсь час, зазвичай кілька хвилин, живу сторінку можна буде подивитися за адресою, вказаною на вкладці Settings > Pages в налаштуваннях репозиторію. Наприклад, ось посилання на живу версію для цього репозиторію

<https://goitacademy.github.io/vanilla-app-template/>.

Якщо відкриється порожня сторінка, переконайся, що у вкладці Console немає помилок пов'язаних з неправильними шляхами до CSS та JS файлів проекту (****404****). Швидше за все у тебе неправильне значення прапора --base для команди build у файлі package.json.

## **Як це працює**

## 

1. Після кожного пуша у гілку main GitHub-репозиторію, запускається спеціальний скрипт (GitHub Action) із файлу .github/workflows/deploy.yml.
2. Усі файли репозиторію копіюються на сервер, де проект ініціалізується та проходить лінтинг та збірку перед деплоєм.
3. Якщо всі кроки пройшли успішно, зібрана продакшн версія файлів проекту відправляється у гілку gh-pages. В іншому випадку, у лозі виконання скрипта буде вказано в чому проблема.

# **.editorconfig**

root = true

[\*]

indent\_style = space

indent\_size = 2

charset = utf-8

trim\_trailing\_whitespace = true

insert\_final\_newline = false

[\*.{json,md,yaml}]

indent\_size = 2

# **.gitignore**

# Logs

logs

\*.log

npm-debug.log\*

yarn-debug.log\*

yarn-error.log\*

pnpm-debug.log\*

lerna-debug.log\*

node\_modules

dist

dist-ssr

\*.local

# Editor directories and files

.vscode/\*

!.vscode/extensions.json

.idea

.DS\_Store

\*.suo

\*.ntvs\*

\*.njsproj

\*.sln

\*.sw?

# **.prettierrc.json**

{

"printWidth": 80,

"useTabs": false,

"semi": true,

"singleQuote": true,

"trailingComma": "es5",

"bracketSpacing": true,

"arrowParens": "avoid",

"proseWrap": "always"

}